Q U A N T A

THE NEWSLETTER OF THE INDEPENDENT QL USERS' GROUP

VOLUME 1 ISSUE 1

FEBRUARY 1984

CONTENTS

INFORMATION ON THE GROUP

Membership of the Group is by subscription to the Newsletter, which is published monthly. Membership details are obtainable from the Group Secretary. Membership of the Group is open to anyone with an interest in the Sinclair QL Microcomputer.

Members requiring assistance with problems related to the QL may call the Secretary. An attempt will be made to put them in touch with a member who can help with the problem.

Workshops will be arranged from time to time in various parts of the country.

A membership list is obtainable from the Secretary.

Please send all contributions for the Newsletter to the Editor.

Acting Chairman
and Newsletter Editor:-

Leon Heller,
8 Morris Walk,
Newport Pagnell.
Bucks. MK16 8QD.
Tel: (0908) 613004

Acting Secretary and
Newsletter Publisher:-

Brian Pain,
24 Oxford Street,
Stony Stratford,
Milton Keynes.
Tel: (0908) 564271

EDITORIAL

Welcome to Quanta, the newsletter of IQLUG, the Independent QL Users' Group.

Since we wanted to get the group going as soon as possible, I have written this first issue myself. This is YOUR newsletter, to be written by YOU, the members of the Group.

My policy will be to publish everything that is sent in, be it good, bad, or indifferent (to coin a phrase), provided that it is not libellous, defamatory, or obscene.

Even if you do not feel like writing anything, let me have your requests for the type of articles you would like to see in Quanta. I will publish these requests, and hopefully, someone will come up with the goods.

Also write in or 'phone if you are having problems with your system. I will be including a Problem Section in future issues.

Complimentary copies of Quanta will be going to Sir Clive Sinclair and Mr. Nigel Searle of Sinclair Research Ltd., PSION, the Amateur Computer Club, PCW, BYTE, Sinclair User and the Amateur Computer Club of New Jersey (USA). That way we preserve our independence, and make sure that the people that matter know of our existence.

Please have a look at the proposed Constitution and let me have your views. It is based upon that of the Amateur Computer Club. We will be having it looked over by a solicitor, to make sure that there are no loopholes. If any of you are legally qualified perhaps you could let me know what you think of it.

As soon as is practicable we intend to organise a workshop. combined with an inaugural General Meeting. We will then be able to hold a formal election for officers of the Group, and get the Constitution accepted. In the mean time, Brian Pain and I will run things. We will be standing for election at the inaugural meeting.

You might like a bit of information about Brian and myself. Brian lectures in accountancy at a College of Higher Education, and I work for myself as a consultant. We formed what is now the National TRS-80 and Genie Users Group some five years ago, and thought that we would organise something along the same lines when the QL came along. I have quite a lot of experience of the 68000 and 68008, as I have a couple of 'home-made' small systems using these processors. I have a 68000 cross-assembler running on one of my other systems, which enables me to write 68000 code and run it on the 68000 or 68008.

I hope you like IQLUG.

Leon Heller

PROPOSED CONSTITUTION FOR IQLUG

1. TITLE

1.1 The name of the Group will be the Independent QL Users' Group (IQLUG), hereinafter referred to as the Group.

2. OBJECT

2.1 The object of the Group will be the dissemination of information pertaining to the Sinclair QL Microcomputer.

3. AFFILIATION

3.1 The Group will be affiliated to the Association of Computer Clubs.

4. MEMBERSHIP

4.1 Any person interested in the operation and use of the Sinclair QL Microcomputer shall be eligible for membership.

5. COMMITTEE AND OFFICERS

5.1 The business of the Group shall be managed by a committee.

5.2 The officers of the Group shall be the Chairman, Secretary and Treasurer.

5.3 The Committee shall consist of the officers of the Group, and not more than, three members elected at the Annual General Meeting. The Committee shall have the power to co-opt additional members as they deem necessary. Such co-opted members shall not be eligible to vote at meetings of the Committee. The Committee may appoint its members to the posts of Membership Secretary, Software Librarian and Newsletter Editor.

5.4 Committee members whose duties involve them in a considerable amount of work will be paid on a Monthly basis according to the time expended upon Group business. Officers of the Group will also be able to claim out of pocket expenses from the Group.

5.5 The members of the Committee shall hold office from the date of appointment until the next Annual General Meeting, and shall be eligible for re-election.

5.6 Should any member of the Committee resign or cease to act during the life of the Committee, the Committee may appoint a member of the Group to fill the vacancy.

5.7 The Committee shall have the power to fix the date of its meetings and to make rules regarding the transaction of business at such meetings. Special meetings of the Committee shall be called at the discretion of the Chairman, or on receipt by the Secretary or Chairman of a written request from not less than two members of the Committee.

5.8 Five members of the Committee shall constitute a forum.

5.9 Any resolution passed by a majority of the members present and voting at a meeting of the Committee shall be the decision of the Committee. In the event of an equality of voting the presiding member shall have an additional or casting vote.

6. SUBSCRIPTIONS

6.1 The Group's year of accounts shall end on (to be decided).

6.2 All members shall pay an annual subscription at rates to be fixed by an Annual or Special General Meeting of the Group. No member shall be entitled to the privileges of the Group until his or her subscription is paid.

6.3 The Group may terminate the membership of a member whose conduct is proved to their satisfaction to be detrimental to the interests of his or her fellow members. Any charge against a member must be made in writing. An appeal against a decision of the Committee may be made to the Annual General Meeting.

7. MEETINGS

7.1 The Annual General Meeting of the Group shall be held in each year within three months of the end of the Group's year of accounts. Notice of resolution and amendments to the constitution shall be forwarded to the Secretary not later than (to be decided) each year. Notice of the Annual General Meeting and agenda shall be circulated to all members not later than 21 days before the date of the Meeting.

7.2 A Special General Meeting of the Group shall be held whenever the Committee think it necessary or whenever 25 or More members request by individual demand in writing; setting out business to be discussed. and delivered to the Secretary. Not less than 21 days notice of any Special General Meeting shall be given.

7.3 A new member present and eligible to vote at an Annual or Special General Meeting shall be given one vote, and when the votes are equal, then the presiding member shall have an additional or casting vote.

7.4 Any member entitled to vote on resolutions and amendments to the Constitution, but unable to be present at an Annual or Special General Meeting is entitled to a postal vote. Postal votes are only valid when sent to the Secretary on the appropriate Postal Vote Form, and which arrive at least 72 hours before the date of the Annual or Special General Meeting.

8. AUDITOR

8.1 An auditor shall be appointed at the Annual General Meeting. He or she shall not be a member of the Group.

9. SECRETARY

9.1 The Secretary shall cause adequate records to be kept of all proceedings of the Committee and of General and Special Meetings. He or she shall, on occasions in the execution of his office but under the superintendence and direction of the Committee. and between meetings of the Committee, be responsible for the conduct of the business of the Group in consultation with the Chairman.

10 TREASURER

10.1 The Treasurer shall take charge of the funds and all receipts of the Group and shall pay all demands under the authority of the Committee. He shall render full and complete accounts at each audit, and whenever required to do so by resolution of the Committee. The Treasurer shall be responsible for maintenance of records of plant and equipment belonging to the Group.

11. INTERPRETATION

11.1 In, these rules, the following words and expressions shall have the meanings which follow, unless such meanings are inconsistent with the subject of the context:

11.1.1 Words importing the singular shall include the plural and vice-versa.

11.1.2 Words importing the masculine gender shall include the feminine.

11.2 In any case of doubt as to the meaning of a rule or its applicability to a particular matter, the Committee shall have the power to decide the issue.

11.3 Nothing in these rules is to be interpreted as conflicting with the rules or constitution of any body or Association to which the Association is affiliated.

12. AMENDMENT OF THE CONSTITUTION

12.1 The Constitution shall not be added to nor amended, nor rescinded in whole or in part, unless details of the proposed changes have been included in the Agenda paper of an Annual General Meeting, and unless the proposal shall have been adopted by at least two-thirds of the members present and voting at such a meeting. Provided that nothing in these rules, certain or implied, shall authorise the amendment or rescission of Section 14.2 which shall not be subject to variation or revocation whatsoever.

13. SURRENDER OF GROUP PROPERTY

13.1 Any member of the Group who has the custody of any software, books, documents, records, property, or monies belonging to the Group shall on request surrender them to the Committee.

14. WINDING-UP

14.1 28 days of any proposal to wind up the Group shall be given in writing to the members of the Group and a Special General Meeting shall be called to consider such a proposal. To be effective. a formal resolution to wind-up the Group must be carried by a vote of at least two-thirds of the members present and entitled to vote.

14.2 Upon the dissolution or winding-up of the Group, the property of the Group shall be disposed of at open auction and the proceeds, together with the pecuniary assets of the Group, shall be donated to the British Red Cross Society.

INTERESTING SUPERBASIC FEATURES

SuperBASIC appears to be substantially upwards compatible with Microsoft's MBASIC and BBC BASIC. That is, many existing programs written in these BASIC dialects should run on the QL with relatively few changes. SuperBASIC, however, offers many advanced additional features, such as powerful constructs to help the programmer to write well-structured, modular programs. Typical of such features is the REPeat keyword which is used in conjunction with the EXIT statement in situations like:-

```
100 REPeat set_total
110       INPUT “Number?”,number
120       IF number = -999 THEN
130             PRINT “Total = ” ,total
140             EXIT set_total
150       ELSE
160             total = total + number
170       END IF
180 END REPeat set_total
```

GOTO and GOSUB are provided for programmers who like 'spaghetti bowl' programming, but should be used sparingly. if at all.

Here are some of the more interesting key words, noted from a rapid perusal of the provisional users guide.

BAUD sets the baud rate for the two serial ports. Both ports, and presumably the input and output channels on each port, must have the same baud rate. It should be possible to switch baud rates 'on the fly’ so this might not be too much of a problem (my thanks to Peter Whittle of the ACC for the above suggestion).

BEEP produces sound. Parameters can be specified for the duration, pitch, pitch range, wrap-around and 'fuzzyness'. I hope Sir Clive has provided us with a decent amplifier and loudspeaker, as I do not believe the sound is output to the TV.

WINDOW fills a block of a specified size and shape, at a specified position on the screen, within the current window.

COPY transfers data from a channel or device to another channel or device, until an end of file marker is reached, or a timeout occurs. It appears as if COPY allows one to transfer data from anywhere to anywhere else! For instance, it should be possible to transfer input from one of the serial ports to a Microdrive file, simultaneously displaying it on the screen and outputting it to the other serial port.

CSIZE sets the width and height of each character cell on the display. The character will be adjusted to fit this cell size.

DATE$ returns the date and time. The QL has a battery backed-up real-time clock. I wonder if files can be time and date stamped?

Defined functions and procedures are available. Formal parameters (those specified in the function or procedure call) and parameters defined as LOCAL within the function or procedure, have no effect on identifiers with similar names used in other parts of the program. The use of procedures allows the programmer to define new keywords. Is recursion allowed? I hope so.

EXEC loads a sequence of programs and executes them in parallel or concurrently. 'Pipes' can be set up between programs so that one program or device can accept as input the output from another program. This is similar to the use of pipes in the UNIX operating system.

EXIT allows one to jump out of a repeat structure and continue processing.

MERGE enables one program to be loaded and merged with another program already in RAM.

PAN enables the contents of the current window to be scrolled horizontally to the left or right by a specified number of pixels. This should be very useful for people writing game programs.

SBYTES saves blocks of memory to a QL device. There should be a corresponding load command, but I could not find it.

SCALE allows the graphics scale factor to be altered. To spread a graph to fill the frame, for instance.

SCROLL scrolls the current window up or down.

SELECT is similar to the CASE statement in the C or Pascal languages.

This is probably the best BASIC available on ANY computer!

Leon Heller

WHAT IS C?

Like a lot of other good ideas. such as the transistor, Unix and non-metric multi-dimensional scaling, the C programming language emanated from the Bell Research Laboratories, usually referred to as Bell Labs., in the States.

One of the Bell Labs. programmers designed a language called B, loosely based on a British language called BCPL, which incidentally, is available on the Beeb. The first version of the Unix operating system was written in B. B was subsequently enhanced, and became C. Unix was subsequently re-written in C, and it has remained the main programming language used under Unix ever since.

Unlike BASIC, which is usually implemented as an interpreter, C is a compiled language. That is, a C program is taken by the C compiler. and the program as a whole is translated into executable machine language, which can then be run as a stand-alone program, without the original source code or compiler. Most C compilers produce assembly language, in fact, which is then assembled into machine language by an assembler. Although BASIC compilers do exist, BASIC programs are usually run by the BASIC interpreter which takes the program a statement at a time, checks the statement for errors, and then executes the statement, usually by a machine language subroutine. Because the interpreter has to check each statement while the program is running, and work out where GOTOs have to go to, and make sure that a GOSUB goes to the right place, BASIC programs run rather slowly compared to the machine language produced by a compiler. They also take up more memory, as the original program, and the interpreter have to be in memory at the same time. Compiled programs, on the other hand, only require enough memory to hold the resultant machine language. The compiler itself is usually a rather large program, but this only needs to be in memory during the compilation, process.

Incidentally, there are compilers, such as UCSD Pascal, that compile to an intermediate form of code, that is then executed by an interpreter.

Let us now look at a simple C program, and see how it is compiled and executed.

The simplest C program is probably the following:-

```
main(){
     printf(“Hello world!\n”);
}
```

This is the first example in the Kernishan and Ritchie classic book on the C language, and merely prints 'Hello world!' on the output device. One of my CP/M C compilers is provided with an example that prints 'Hi sailor! New in town?'. This, presumably, is an American version of 'Hello sailor!'.

With most C compilers, one would first create a text file containing the above program with a text editor, and save it to disk (or Microdrive, in the case of the QL) under the name 'HELLO.C'. Typing C HELLO would then load the C compiler, which would proceed to compile the program, and produce another, assembly language, file called HELLO.ASM This intermediate assembly language file would then be assembled into executable machine language or object code by typing ASM HELLO. One can then run and test the program just by typing HELLO. Although details of the QL C compiler have not yet been released. it will probably be used as I have just described.

A C program is built up out of functions, which correspond roughly to subroutines in BASIC. A program must consist of at least one function called main. Parameters are passed to functions by enclosing them in parentheses following the function name. If there are no parameters. the parentheses are left empty, as in the HELLO program. A statement, such as printf('Hello world!'), must be terminated by a semi-colon. More than one statement may appear on a line. A function can return a value, just like procedures in QL SuperBASIC. Most C compilers are provided with a function library, for things like I/O, string handling, mathematical functions, etc. One can also form one's own library, and gradually build up a library of useful functions that greatly reduces the time spent in writing a program. I have developed a function library for my CP/M system that contains functions to do thing's like clear the screen, generate random numbers, simulate all the BASIC string handling commands, print text anywhere on the screen. etc.

You might wonder why I had to write my own functions to do things that are usually provided as part of a BASIC interpreter, for instance. Well, C is a relatively low-level, high-level language. Some people refer to it as a high-level assembler. Unlike most compiled languages, C is very close to the actual machine, and is intended for systems programming; writing assemblers, text editors, compilers etc. C compilers are usually written in C! In fact. I have heard that PSION who wrote all the software for the QL, wrote most of it in C, using a 68000 cross-compiler running on a VAX. C is a very ‘portable’ language, in that programs written for one machine will usually work on another machine, with few if any chances being necessary.

Here is a another, larger program, which illustrates most of the features of C. Although it was written for the C/80 compiler on the TRS-8O Model II under CP/M, it should work almost 'as is' on the QL, when the C compiler comes along. All that will probably need changing will be the two little assembly language routines at the end.

```
/*          * * * * * * * * * * * * * * * * *
            *                               *
            *           TERM                *
            *                               *
            *                               *
            * * * * * * * * * * * * * * * * *
```

Smart terminal program for the TRS-80 Model II written in C for the C/80 compiler.

Version, 1.1 - 18/8/83.

Written by Leon Heller.

Modify serin, and serout functions for your particular hardware configuration.

Baud rate must be set using a CONFIG utility, or whatever is supplied for your system. I shall be adding baud rate selection to this program at a future date.

Commands are prefixed by <ESCAPE:> and are as follows:-

<Q>UIT- Leave TERM and return to CP/M
<O>PEN - Open buffer
<C>LOSE - Close buffer
<S>AVE - Save buffer to disk
<T>RANSMIT - Transmit file
<H>ELP - Display these commands

Commands maw be upper- or lower-case.

I cannot claim that this program is particularly well written but it was my first attempt at using C. Ideally, the buffer array space should be allocated at run-time, but I have not worked out yet how to do this!

Functions cls, kbrd, getkey, gets, puts and gotoxy are in an external library called mylib.c, included on this disk.

Using the Macro-80 LIB80 Library manager program, the C/80 library, I/O library and mylib are linked to form lib.rel.

Using C/80, and Macro-80 the program is compiled, assembled, and linked by the following SUBMIT file:-

```
C-M $1 >ERRORS
TYPE ERRORS
M80 $1=$I
L80 $1,LIB/S,$1/N/E
```

by typing SUBMIT C TERM at CP/M command level.

```
*/

£include <printf.h>

/* global definitions including assembler equates */

/* TRS-80 Model II SIO ports */
£define ACMD 0F6H
£define ASTAT 0F6H
£define ADATA 0F4H

£define EOF -1
£define CR 13
£define LF 10
£define ESC 27
£define YES 1
£define NO 0
£define SIZE 32767
£define XOFF 19
int FLAG, n, i;
char buff[SIZE];
char c;

main(){
	FLAG = NO; /* reset buffer flag */
	cls();
	print(30,1,"* * * * * * * * * * * * * * * * * ");
	print(30,2,"*                               *");
	print(30,3,"*              TERM             *");
	print(30,4,"*                               *");
	print(30,5,"* * * * * * * * * * * * * * * * * ");
	while(1){
		c=kbrd ();
/* if <ESCAPE> pressed get command */
		if (c==ESC) cmd();
		else{
			if (c!=0) serout(c);
			c=serin();
			if (c!=0){
				dsply(c);
				if (FLAG==YES) {
					buff[n]=c;
					n++;
```

```
                                        if (n==SIZE) {
                                                printf(“\n<BUFFER FULL>\n”);
                                                FLAG=NO;
                                        }
                                }
                        }
                }
        }
}

/* set command from user */
cmd(){
        char c;
        c = getkey() & 0X5F; /* convert to upper case */
        switch (c) {
                case 'Q':quit();
                break;
                case 'O':open();
                break;
                case 'C':close();
                break;
                case 'S:save();
                break.;
                case 'T':xmit();
                break;
                case 'H':help();
                break;
                default:return;
                break;
        }
}

/* quit program and return to CP/M */
quit() {
        cls();
        exit();
}

/* open buffer. If buffer already open do nothing */
open() {
        if (FLAG==NO){
                FLAG=YES;
                n=0;
                printf(“\n<BUFFER OPEN>”);
                return;
        }
        else printf(“\n<NOTHING DONE>”);
        return;
}

/* Close buffer. If buffer already closed do nothing */
close() {
        if (FLAG==YES) {
                FLAG =NO;
                printf(“\n<BUFFER CLOSED>”);
```

```
	}
	Else printf("\n<NOTHING DONE>");
}

/* Save buffer contents to disk. If buffer still open do nothing */
save() {
	char c, fname[12];
	int char, i;
	if (FLAG==YES) {
		printf("\n<NOTHING DONE>\n"):
		return;
	}
	printf("\n<SAVE BUFFER>\n") ;
	printf("FILENAME: ");
	gets(fname);
	chan=fopen(fname,"w");
	If (chan==0) {
		printf("<OPEN ERROR:>\n");
		return;
	}
	for (i=0;I<n;i++) {
		c=buff[i];
		putc(c,chan);
	}
	fclose(chan);
}

/* Transmit file to remote system */
xmit() {
	chan c, fname[12];
	in chan;
	printf("\n<TRANSMIT FILE>"):
	printf("\nFILENAME: ");
	gets(fname);
	chan=fopen(fname,"n");
	if (chan==0) {
		printf("<OPEN ERROR>\n");
		return;
	}
	while (c!=EOF) {
		c=getc(chan);
/* send a few nulls if new line */
		if(c=="\n" {serout(CR);serout(0);serout(0):serout(0);}
		else serout(c);
	}
	fclose(chan);
}

/* Display Help menu */
help() {
	printf("\n<C>LOSE\n<H>ELP\n<O>PEN\n<Q>UIT\n");
	printf(" <S>AVE\n<T>RANSMIT\n");
}
```

```
/* assembler routine for serial output */
serout(c) char c;{
	c;
	£asm
	.Z80
  S3:XOR	A
	OUT	(ACMD),A
	IN	A,(ASTAT)
	BIT	2,A
	JR	Z,S3
	LD	A,L
	AND	7FH
	OUT	(ADATA),A
	.8080
	£endasm
}

/* assembler routine for serial input */
serin() {
	£asm
	.Z80
  S1:XOR	A
	OUT	(ACMD),A
	IN	A,(ASTAT)
	BIT	0,A
	JR	NZ,S2
	XOR	 A
	JR S4
  S2:IN A,(ADATA)
	AND	7FH
  S4:LD	L,A
	.8080
	£endasm
}

/* display string on screen at (x,y) */
print(x,y,s) int x,y;
		char *s;{
	gotoxy(x,y);
	puts(s);
}
```

Leon Heller

AN INTRODUCTION TO THE 68008

The 68008 Programming model is described in Fig. I, and consists of 17 32-bit registers, a 32-bit program counter, and a 16-bit status register. The first eight registers (D0 - D7) are used as data registers for bit, byte (8-bit), word (16-bit) and long word (32-bit) operations. The second set of seven registers (A0 - A6) and the system stack pointer (A7) may be used as stack pointers and base address registers. A0 - A6 May be used for word and long word operations. All 16 registers may be used as index registers. The system stack pointer may be either the supervisor stack pointer (SSP) or the user stack pointer (USP), depending upon the status of bit 13 (S bit) in the status register.

The status register (Fig. 2) may be thought of as consisting of two bytes, a user byte, and a system byte. The user byte contains five bits which define the carry (C), overflow (V), zero (Z), negative (N) and extend (X) condition codes. The system byte contains five bits which define the current interrupt priority (four levels of interrupt), whether the processor is in Trace mode, and/or the supervisor or user state. In Trace mode, the processor will go into an exception processing state (like a software interrupt) after every instruction. This makes the writing of single-step debugging utilities very easy. Hardware single-stepping is quite easy to achieve, also.

The program counter is a 32-bit register, but only 20 bits are available in the case of the 68008, giving 1 Mb of address space. The forthcoming 68020 32-bit processor will have a full 32-bit program counter, giving 4 gigabytes of address space! They had better start producing 1 Mbit RAMs as soon as possible!

Most instructions operate on bytes, words, and long words.

There are 14 addressing modes available:-

Register Direct

Data Register Direct
Address Register Direct

Absolute Data

Absolute Short
Absolute Long

Program Counter

Relative with Offset
Relative with Index and Offset

68008
PROGRAMMING MODEL

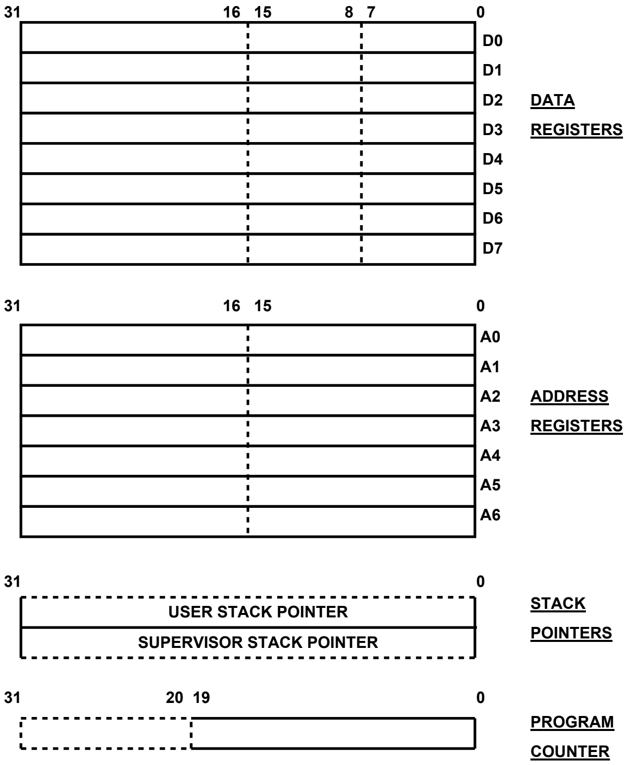

Fig. 1

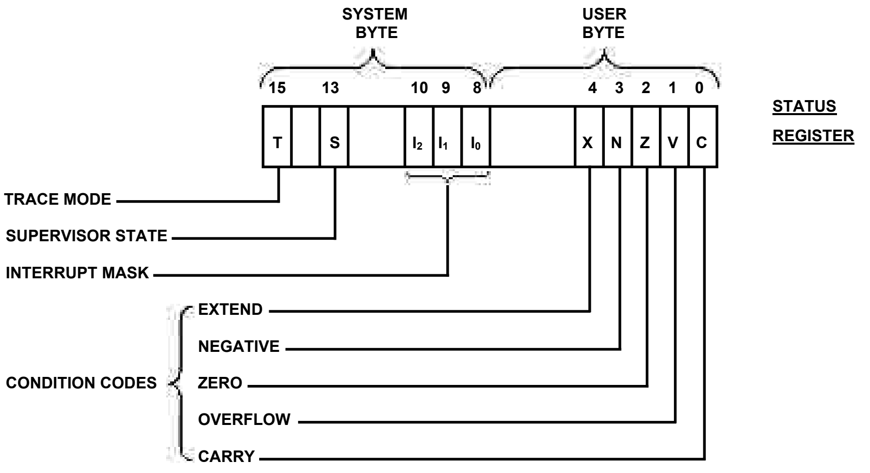

Fig. 2

Register Indirect

Register Indirect
Postincrement Register Indirect
Predecrement Register Indirect
Register Indirect with Offset
Indexed Register Indirect with Offset

Immediate Data

Immediate
Quick Immediate

Implied

Implied Register

The instruction set was designed around a relatively small number of mnemonics, to make things easier for the programmer. However, the large number of addressing modes, which apply to most of the instructions (what is termed an orthogonal instruction set) mean that there is a total of over 1000 useful instructions! These include signed and unsigned multiply/divide, quick arithmetic operations (with an 8-bit quantity as part of the op. code), and BCD arithmetic. If that is not enough, certain op. codes initiate exception processing, enabling you to define your own instructions and simulate their operation in software.

If you value your sanity, do not try hand-assembling 68000 code! An assembler is essential if the program is more than about six lines long, unless you you are one of those gentlemen in the Tefal commercials.

Leon Heller

This article was based upon an article previously published in NATGUG News, the newsletter of the National TRS-80 and Genie Users' Group.

32 BITS INDEED! WHO DO SINCLAIR THINK THEY ARE KIDDING?

In the advertisements for the QL, Sinclair make the claim that the MC68008 processor is a 32-bit device. Motorola, who make the MC68008, and who presumably know more about it than Sinclair's advertising agency, call it a 16-bit microprocessor with an 8-bit data bus. If the QL is a 32-bit system, the ZX81 could be described as a 16-bit machine, since it has a number of 16-bit wide registers. Granted, the MC68008 has 17 32-bit registers. which entitles one to refer to it has having a 32-bit architecture, but the all-important Arithmetic Logic Unit or ALU, is only 16 bits wide. Motorola are working on a true 32-bit processor, the MC68020, which is due to be sampled later this year. I would imagine that Sinclair have already been reported to the Advertising Standards Authority over the aforementioned advertisement. I have half a mind to write to them myself!

Leon Heller